DTU Library

Capacity Extension of Software Defined Data Center Networks With Jellyfish Topology

Mehmeri, Victor; Vegas Olmos, Juan José; Tafur Monroy, Idelfonso

Publication date:
2015

Document Version
Publisher's PDF, also known as Version of record

Capacity Extension of Software Defined Data Center Networks with Jellyfish Topology

Victor D. Mehmeri*, J.J. Vegas Olmos, I. Tafur Monroy
Technical University of Denmark, Department of Photonics Engineering, Ørsted Plads, Building 343, Kgs. Lyngby 2800, Denmark
*Author email address: *vime@fotonik.dtu.dk*

Abstract: We present a performance analysis of Jellyfish topology with Software-Defined commodity switches for Data Center networks. Our results show up to a 2-fold performance gain when compared to a Spanning Tree Protocol implementation.
OCIS codes: (060.4256) Network optimization; (060.4250) Networks

1. Introduction

Big Data and Cloud applications are experiencing an exponential growth rate which has been forcing Data Center enterprises to evolve their infrastructure in order to meet the higher bandwidth demand and QoS requirements of these services. Expanding network capacity has hence become a major concern, and to tackle this challenge, several solutions such as those employing special network topologies and routing structures have been proposed [1-4]. As argued in [5], however, none of these architectures address the incremental expansion problem, i.e., the ability to add new servers and network capacity while preserving structural properties. In addition, some of the proposed solutions require complex routing and/or a significant amount of manual configurations, driving up the cost of deployment.

The Software Defined Networking (SDN) paradigm has also been largely explored in academic research and widely adopted in Data Center networks in the last few years due to its improved flexibility and manageability, and also for its potential for cost reduction. There still is, despite of that, a big concern among enterprises about SDN being slow and not scalable due to the overhead of a centralized control. However, it has been shown on a previous work that SDN can outperform a traditional layer 2 Spanning Tree Protocol in networks with highly interconnected topologies [6], and it has also been shown that most of the scalability issues of SDN can be addressed without losing its benefits [7].

The goal of this paper is to evaluate how a Jellyfish topology with bare-metal, OpenFlow-enabled switches using a centralized open-source SDN controller performs when compared to the same topology running Spanning Tree Protocol. We thus attempt to verify the validity of such architecture as a flexible, scalable and low cost approach for Data Center networking.

In Section 2, we provide a background introduction. Then, in Section 3, we discuss cost, performance and scalability aspects. We describe in Section 4 the evaluation testbed for measuring the performance of an SDN implementation against Spanning Tree Protocol, regarding network throughput, delay, jitter, and loss rate. Finally, in Section 5 we present and discuss the results obtained.

2. Background

Layer 2 switching can be a reliable strategy for data center networks with commodity hardware. A typical L2 switching protocol, Spanning Tree Protocol (STP), is used to ensure loop-free communication paths. However, it causes the switches to block redundant ports in order to create a single path tree, which is often a suboptimal routing path in highly interconnected topologies, resulting in loss of bandwidth and increased end-to-end latency. STP is therefore incapable of leveraging path redundancy in a network, and has been shown to perform worse than an SDN implementation with an out-of-the-box open source controller in interconnecting schemes with redundant paths such as Torus and Hypercube [6].

A more traditional and widely deployed network topology in Data Centers is Fat-Tree, which is capable of delivering high bisection bandwidth due to its path multiplicity and special routing algorithm [1]. On the other hand, Jellyfish, a random graph based topology, has been shown to be more cost-efficient than a Fat-Tree using the same equipment, supporting as many as 25% more servers at full capacity [5]. In addition, Jellyfish interconnect provides good ability of incremental expansion and allows heterogeneity in switches port count, a good advantage in terms of flexibility.